DOCUMENT RESUME

ED 348 987 IR 015 686

AUTHOR Reed, W. Michael; Liu, Min
TITLE The Comparative Effects of BASIC Programming versus HyperCard Programming on Problem-Solving, Computer Anxiety, and Performance.
PUB DATE Mar 92
NOTE 25p.; Paper presented at the Annual Conference of the Eastern Educational Research Association (Hilton Head, SC, March 1992).
PUB TYPE Reports - Research/Technical (143) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Analysis of Variance; *Anxiety; Higher Education; Hypermedia; *Instructional Effectiveness; *Performance; Predictor Variables; Pretests Posttests; *Problem Solving; *Programing; Programing Languages; Regression (Statistics)
IDENTIFIERS *BASIC Programing Language; Computer Anxiety; *HyperCard

ABSTRACT

This study compared the effectiveness of the BASIC programming language and the HyperCard authoring language on promoting problem-solving skills and the reduction of computer anxiety. Two groups of students participated in the study. One group (N=13) received instruction on BASIC and the other (N=8) on HyperCard. Students in both groups responded to the same 61-item problem-solving instrument and 20-item computer anxiety instrument during the first and 12th meetings. During the second through the ninth meetings, they received instruction on one of the programming languages; during the 10th meeting they took either the BASIC or the HyperCard test; and during the 11th meeting, the evaluated tests were returned and answers were explained. Analyses of the data indicated that the problem solving skills of the BASIC group improved from the beginning of the treatment to the end whereas the problem solving skills of the HyperCard group did not; computer anxiety decreased for both groups; the HyperCard group outperformed the BASIC group on both the programming and debugging exercises; and a negative relationship between computer anxiety and debugging emerged when dealing only with the BASIC group. It is concluded that, while HyperCard is superior to BASIC for software construction, BASIC appears to be the language of choice when the purpose is to develop problem-solving skills. The programming portions of the BASIC and HyperCard tests and exercises for debugging in the two languages are appended. (Contains 40 references.) (BBM)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

☐ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

ED348987

The Comparative Effects of BASIC Programming Versus HyperCard Programming on Problem-Solving, Computer Anxiety, and Performance

by

W. Michael Reed & Min Liu
West Virginia University
404-B Allen Hall
Morgantown, WV 26506

Paper Presented at the Annual Meeting of
Eastern Education Research Association
Hilton Head, South Carolina
1992

112015686

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Min Liu

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

BEST COPY AVAILABLE

The Comparative Effects of BASIC Programming Versus HyperCard Programming on Problem-Solving, Computer Anxiety, and Performance

RESEARCH FRAMEWORK

One of the most common research focuses in instructional computing during the last decade has been the learning of programming languages for the development of problem-solving skills (i.e., Clements, 1986; Dalbey & Linn, 1985; Reed & Palumbo, 1992). Although findings have been somewhat inconclusive, there have consistent trends in later and lengthier studies that learning such languages will enhance problem-solving skills (see Magliaro & Burton, 1987/1988). Although a plethora of studies on programming languages has emerged, very little research on authoring languages such as HyperCard has been conducted.

Even though a search of ERIC documents from January 1982 to September 1991 resulted in identifying 99 articles specifically about HyperCard, about 90 percent of them were on the educational applications of HyperCard: (a) personal experiences of using HyperCard or developing HyperCard-based courseware (i.e., Bowers & Chia, 1990; Hopper, 1990; Percival 1990); (b) discussions on how to use HyperCard (i.e., Duhrkopf, 1988; Harris & Cady, 1988; Vacarro, 1988); (c) explorations of the potential of HyperCard (i.e., Camp & Cogan, 1988; Campbell, 1989; May, 1990; McDowell, 1989) and (d) comparisons between HyperCard and other authoring tools (i.e., Brader, 1990; Harris & Cady).

The different content-areas in which HyperCard has been applied include (a) language and learning (i.e., Campbell, 1989; DiPardo & DiPardo, 1990; May, 1990); (b) history (i.e., Crane & Mylonas, 1988); (c) science (i.e., Barba & Merchant, 1990; Hall, Thorgood, Hutchings, & Carr, 1989; Miller & Radziemski, 1988); (d) second language learning (i.e., Ashworth & Stelovsky, 1989; Underwood, 1989); (e) library science (i.e., Bourne, 1990; Holleman, 1990; Neuman, 1991); (f) theatrical performance (i.e., Friedlander, 1989); (g) computer science (i.e., Gustafson

& Reeves, 1990; Percival, 1990); and (h) social studies (i.e., Horton, Boone, & Lovitt, 1990). Categories other than content-area were (a) multimedia and hypermedia (i.e., Ashworth & Stelovsky; Campbell; Jonassen, 1989; Lanz & Roselli, 1991; Tripp & Roby, 1990) and (b) learning disabilities (i.e., Horton, Boone, & Lovitt).

HyperCard has been found to be an excellent tool for creating prototypes of some more complicated and comprehensive projects (i.e., Boyle & Snell, 1990; Duhrkopf, 1988; Harris & Cady, 1988; Jonassen, 1989, 1990; Monk, 1990). Also, it has been used in training and teaching and has received positive reactions (i.e., Dufresne, Jolin, & Senteni, 1990; Tucker & Dempsey, 1990). Likewise, it has been used extensively and effectively in illustrating instructional design principles and multimedia and hypermedia theories and concepts (i.e., Boyle & Snell; Gustafson & Reeves, 1990; McKnight, Dillon, & Richardson, 1990).

But, none of these works has addressed the problem-solving potential of learning authoring languages such as HyperCard, a computer use that has been the dominating, computer-related research line for years. Despite the proven importance of programming languages' role in developing problem solving skills, the creation of authoring languages such as HyperCard has led some computer educators to question the teaching of such languages as BASIC at all. While most computer educators would concur that using BASIC to create any complex software is unwise—certainly when comparing it to the easier and multi-featured authoring language such as HyperCard—choosing HyperCard over BASIC, when the purpose of the instruction is to promote problem-solving skills, may not necessarily be the sound decision. That is, if the purpose of teaching a language is to enhance problem-solving skills, choosing a more flexible authoring language may not promote the development of problem-solving skills as much as a programming language that requires the developer to work with each command, rather than with HyperCard objects that simplify the construction of programs.

The few HyperCard-based studies that exist focus on using already-developed software, rather than programming in HyperCard. For example, Lanza and Roselli (1991) investigated the effects of a structured approach versus a hypertextual approach for teaching Pascal. The study,

with Information Science undergraduates, involved comparing the effect of a HyperCard-based, non-linear and less structured program versus a linear and more structured CAI program on the learning of Pascal. Although there were no significant performance differences, 63 percent of the CAI group considered the structure program to be boring whereas 76 percent of the HyperCard group considered the hypertextual approach to be stimulating.

Tripp and Roby (1990) investigated a less structured versus a more structured hypertext environment using HyperCard in terms of their relative effects on the learning of Japanese by graduate and undergraduate students. More structured involved a visual that suggested the structure of the database whereas less structured involved a graphical background that did not reflect the structure of the database. More structure also involved advance organizers describing the structure of the lexicon whereas less structured involved presenting a paragraph on the Japanese language. There were no significant main effects, but there was a significant interaction between visual metaphor and advance organizer with those students receiving only one structured device performing better than those receiving two.

In another study, Horton, Boone, and Lovitt (1990) used HyperCard-based, hypertext software to increase textbook comprehension and compared it to when the students simply read the textbook. The computer treatment was significantly more effective. A fourth study, one by Collourn and Cockerton-Turner (1990), found that pairs versus individuals performed better when attempting a computer-presented physics problem.

There is obviously little research on HyperCard to-date, and the few briefly described above do not represent any consistent focus. A critical area not represented by any of the research is the effect of HyperCard on developing problem solving skills, an area that has been valued and heavily researched during the last decade.

PURPOSE OF STUDY AND RESEARCH QUESTIONS

The purpose of this study was to compare the effectiveness of the BASIC programming language versus the HyperCard authoring language on promoting problem-solving skills and the

reduction of computer anxiety as well as to determine the relationships between (a) problem-solving and computer anxiety, (b) problem-solving and programming, (c) problem-solving and debugging, (d) computer anxiety and programming, and (e) computer anxiety and debugging.

Specifically, the research questions were (a) what are the comparative effects of BASIC versus HyperCard on problem solving and computer anxiety? (b) what effect does language have on language competency measures (debugging and programming)?, (c) what is the relationship between problem-solving and (1) computer anxiety and (2) language competency as measured via programming and debugging exercises? and (d) what is the relationship between computer anxiety and language competency as measured via programming and debugging exercises?

DESIGN OF THE STUDY

Sample

Two groups of students participated in this study. One group (N = 13) received instruction on the BASIC programming language. The second group (N = 8) received instruction on the HyperCard authoring language. The instruction for each language lasted six weeks with two weekly meetings of approximately 90 minutes each. Outside-class assignments required an average of over three hours for each class meeting. Therefore, minimal exposure to each language was approximately 54 hours over a six-week period.

Independent Variable

The independent variable in this study was computer language—either BASIC or HyperCard. BASIC instruction centered on (a) immediate versus delayed BASIC commands; (b) DOS versus Applesoft BASIC commands; (c) static and animated graphics; (d) lo-resolution and hi-resolution graphics; (e) TEXT commands; (f) numeric versus string variables; (g) modularity (subroutines); (h) iteration (FOR/NEXT); and (i) logical constructs (IF/THEN). The HyperCard instruction centered on (a) the browsing environment of HyperCard via the navigation tools; (b) the painting tools; (c) the authoring environment of HyperCard (its five objects: stack, card, background, field, and button); (d) animation; (e) the programming environment of HyperCard

including such commands as go, put, visual effects, show and hide, domenu, repeat, and if/then; and (f) the multimedia capabilities of HyperCard.

Dependent Measures

Problem Solving. The problem-solving instrument was comprised of 61 items taken from the Deduction and Interpretation sections of the *Watson-Glaser Thinking Appraisal* instrument and from the Analysis of Relevant and Irrelevant Information and the Analysis of Attributes sections of the *Ross Test of Higher Cognitive Processes* instrument. The instrument has proved to be valid and highly reliable (Reed & Palumbo, 1987/1988, 1992).

Computer Anxiety. The 20-item computer anxiety instrument was based on a modified version of the Spielberger *Self-Evaluation Questionnaire*. The instrument has proved to be valid and highly reliable (Reed & Palumbo, 1987/1988, 1992).

Programming. As part of the BASIC test, the students were assigned to write a program that (a) asked the user for two numbers; (b) asked the user which mathematical operation—addition, subtraction, multiplication, or division—he or she would like to perform using the two numbers; (c) required an error trap for division by zero; (d) presented the result of the operation; and (e) then gave the user the option to repeat the sequence with two different numbers. The range was 0 points to 35 points. (See Appendix A for the BASIC programming assignment.)

As part of the HyperCard test, the students were assigned to create a stack that consisted of at least four cards with the first card as the title card having one background and the remaining cards having another background. Pictures representing four different objects were to be on the second card. Each object was to be labeled by information in a field. A click on an object would take the user to a description of the object, presented on a card or a field. The program had to have an option to exit at any point, and the instructions for the user were to be clear. The movements between cards were to be accompanied by visual effects. The range was 0 points to 30 points. (See Appendix B for the HyperCard programming assignment.)

Because the programming measure for the BASIC group had a different range (0 to 35)

than the measure for the HyperCard group (0 to 30), scores were changed to percentages, giving a range from 0 to 100.

Debugging. As part of the BASIC test, the students were assigned to debug a program, presented in paper-form, that had 10 errors. They were to identify the 10 errors, explain why each was an error or what would happen if the program were executed without correcting it, and then to correct it. The errors ranged from incorrect syntax, missing commands, excessive numeric variable values, and illogical sequence of commands. The range was 0 points to 30 points. (See Appendix C for the BASIC debugging assignment.)

As part of the HyperCard test, the students were assigned to debug a program in HyperTalk scripting, presented in paper-form, that had 10 errors. They were to engage in the same debugging steps—explain why each was an error or what would happen if the program were executed without correcting it, and then to correct it—as given for the BASIC group. The errors ranged from incorrect syntax to missing commands to illogical commands. The range was also 0 points to 30 points. (See Appendix D for the HyperCard debugging assignment.)

Procedures

During the first class meeting students in both classes responded to the 61-item problem-solving instrument and the 20-item computer anxiety instrument. During the 12th meeting they responded to the same instruments. During the second through the ninth meetings, they received instruction in either BASIC or HyperCard—as explained above under the Independent Variable section. During the 10th meeting, they took either the BASIC or the HyperCard test. During the 11th meeting, the evaluated tests were returned and correct and incorrect answers were explained.

Analysis of the Data

To answer the research question—what are the comparative effects of BASIC versus HyperCard on problem solving and computer anxiety?—two 2 (Course: BASIC versus HyperCard) by 2 (Data Collection: pretest versus posttest) analyses of variance with a repeated measure on one factor (Data Collection: pretest versus posttest) were conducted with problem-solving as the dependent measure in one and computer anxiety as the dependent measure in the

other.

To answer the research question—what effect does language have on language competency measures (debugging and programming)?—two unpaired t-tests were conducted comparing the debugging scores of the HyperCard group versus those of the BASIC group in one procedure and then comparing the programming scores of the HyperCard group versus those of the BASIC group in the other procedure.

To answer the research questions—what is the relationship between problem-solving and (1) computer anxiety and (2) language competency as measured via programming and debugging exercises? and what is the relationship between computer anxiety and language competency as measured via programming and debugging exercises?—regressions were conducted.

RESULTS

Problem Solving

There was a significant main effect of Data Collection on problem solving: $F(1,19) = 8.167$, $p = .01$. That is, the study-participants' problem solving increased from pre-treatment to post-treatment ($X_{pre} = 45.81$; $X_{post} = 48.00$). There was not, however, a significant main effect of Course on problem solving: $F(1,19) = .175$, $p = .68$. There was also not a significant two-way interaction: $F(1,19) = 2.173$, $p = .157$. There was a significant increase in problem solving for the BASIC group: $t(12) = -3.309$, $p = .006$ ($X_{pre} = 44.923$; $X_{post} = 48.00$). There was not, however, a significant change in problem solving for the HyperCard group: $t(7) = -.563$, $p = .591$ ($X_{pre} = 47.25$; $X_{post} = 48.00$). Although there was not a significant main effect of Course on problem solving, the pre-treatment to post-treatment (Data Collection) significant increase was due to the increase for the BASIC group. (See Table 1.)

Computer Anxiety

There was a significant main effect of Data Collection on computer anxiety: $F(1,19) = 54.893$, $p < .0001$. That is, the study-participants' computer anxiety decreased from pre-treatment to post-treatment ($X_{pre} = 47.714$; $X_{post} = 35.476$). There was not, however, a significant main

effect of Course on computer anxiety: $F(1,19) = .478$, $p = .498$. There was also not a significant two-way interaction: $F(1,19) = 2.219$, $p = .153$. There was a significant decrease in computer anxiety for the BASIC group: $t(12) = 5.06$, $p = .0003$ ($X_{pre} = 48.077$; $X_{post} = 37.769$). There was also a significant decrease in computer anxiety for the HyperCard group: $t(7) = 5.486$, $p = .0009$ ($X_{pre} = 47.125$; $X_{post} = 31.75$). These results indicate that the respective treatments were equally effective at significantly reducing the study-participants' computer anxiety. (See Table 1.)

Table 1. ***Means and Standard Deviations of HyperCard and BASIC Groups' Pre-treatment and Post-treatment Scores for Computer Anxiety, Problem Solving, Debugging, and Programming***

	HyperCard n = 8		***BASIC n = 13***		***Both n = 21***	
	Pre	***Post***	***Pre***	***Post***	***Pre***	***Post***
Computer Anxiety[1]	47.125 (8.643)	31.75 (10.553)	48.077 (14.192)	37.769 (11.577)	47.714 (12.133)	35.476 (11.33)
Problem Solving[2]	47.25 (9.192)	48.00 (7.251)	44.923 (5.235)	48.00 (4.865)	45.81 (6.882)	48.00 (5.71)
Debugging[3]		28.375 (2.722)		18.077 (4.192)		22.00 (6.277)
Programming[4]		92.5 (9.754)		82.077 (11.243)		86.762 (11.832)

1 = significant pre-treatment to post-treatment decrease for the HyperCard group, the BASIC group, and both groups combined
2 = significant pre-treatment to post-treatment increase for the BASIC group and both groups combined
3 = significant difference between the HyperCard group and the BASIC group
4 = significant difference between the HyperCard group and the BASIC group

Performance Measures

Programming. The HyperCard group performed better on the programming exercise than did the BASIC group: $t(19) = -2.213$, $p = .039$. The mean for the HyperCard group was 92.75; for the BASIC group, 82.077. (See Table 1.)

Debugging. The HyperCard group also performed better on the debugging exercise than did the BASIC group: $t(19) = -6.612$, $p < .0001$. The mean for the HyperCard group was 28.375; for the BASIC group, 18.077. (See Table 1.)

Correlations

Both Groups Combined. There were significant, high correlations between post-treatment computer anxiety and debugging when factoring in Course: $r = .893$, $p < .0001$. Course had a positive relationship with debugging: $t(20) = 6.531$, $p = .0001$, *beta* = 9.043, *beta weight* = .717. Because the BASIC group was coded as 1 in the regression and the HyperCard group as 2, this finding indicates that the HyperCard group scored higher on the debugging exercise. An unpaired t-test comparing the groups' means, as explained above in the Debugging section, supports this finding. Post-treatment computer anxiety, however, had a negative relationship with debugging: $t(20) = 3.43$, $p = .003$, *beta* = -.209, *beta weight* = -.376. That is, those with relatively lower computer anxiety generally performed better on the debugging exercise, regardless of course grouping. Based on the *beta weights*, course grouping had a stronger relationship with debugging than did computer anxiety. (See Table 1.)

There were no other significant correlations when factoring in Course.

HyperCard Group. There was a significant, high, negative correlation between problem-solving and computer anxiety: $r = -.709$, $t(7) = 2.466$, $p = .049$. That is, for the HyperCard group, those study-participants who had relatively lower computer anxiety generally had relatively higher problem solving scores.

There were no other significant correlations.

BASIC Group. There was a significant, medium, negative correlation between programming and computer anxiety: $r = -.566$, $t(12) = 2.28$, $p = .044$. That is, for the BASIC

group, those study-participants who had relatively lower computer anxiety generally had relatively higher programming scores. There was a significant, medium, negative correlation between debugging and computer anxiety: $r = -.649$, $t(12) = 2.826$, $p = .017$. That is, for the BASIC group, those study-participants who had relatively lower computer anxiety generally had relatively higher debugging scores.

There were no other significant correlations.

DISCUSSION

Problem Solving Skills

The problem solving skills of the BASIC group improved from the beginning of the treatment to the end, whereas the problem solving skills of the HyperCard group did not. This finding supports our earlier notion that HyperCard may simplify programming so much due to its authoring features that thinking processes critical to enhancing problem solving skills may not occur.

Although BASIC and HyperCard are similar in certain ways, a profound difference is ease in use. HyperCard shares certain similar features with BASIC. Each of the five objects in HyperCard (stack, card, background, field, and button) can be compared to the subroutines in BASIC. Each of them has its own characteristics and performs certain functions. Fields are used to store information much like the arrays in BASIC. Like BASIC, HyperTalk (the scripting language for HyperCard) has its control statements, time/date functions, math functions, math operation capability, and file handling capability. However, unlike BASIC, programming in HyperCard is non-linear in nature, although BASIC can be taught in a non-linear fashion. Each of the five objects has its own separate scripts and is independent of one another. Because the objects in HyperCard can be linked in different ways, there are no strict beginning and ending points. But, despite similarities, a fundamental difference is that HyperCard simplifies software construction. (See Henry & Southerly, this issue, for a detailed comparison of the two languages.)

That the BASIC group improved in problem solving is not especially surprising. A large

amount of research to-date has supported this finding. Although many studies exist, a few are virtually identical in design: Reed and Palumbo, 1987/1988; Reed, Palumbo, and Stolar, 1987/1988; and Reed and Palumbo, 1992. In each of these studies, the BASIC participants' problem solving scores increased significantly over the course of nine weeks. One essential difference between this study and those particular three is that this study occurred over a six-week, rather than a nine-week, period. In another study—Palumbo and Reed (1987/1988)—problem solving scores were collected at the four-week point to attempt to determine at what point measurable changes in problem solving might occur. There was not a significant increase from the onset of the study to the four-week mark, indicating that changes in problem solving occurred between the five-week and nine-week points. Because the sample in this study was very comparable to those in the other four, a fairly safe inference is that it takes between five and six weeks—45 to 54 hours—for problem solving skills to be measurably affected.

Computer Anxiety

For both groups computer anxiety decreased. This is not surprising because many studies have shown that computer anxiety can be significantly decreased even after very brief exposures to computer instruction. For example, Overbaugh and Reed (1992) found that computer anxiety was significantly reduced after three weeks, two hours per week, of instruction—only six hours. Studies involving BASIC (i.e., Reed & Palumbo, 1987/1988; Reed & Palumbo, 1992) cite four weeks of instruction as being effective at reducing computer anxiety. This study found that, after six weeks of instruction, computer anxiety was significantly reduced regardless of language. One interesting note is that the BASIC group reduced their anxiety by 10.308 points whereas the HyperCard reduced theirs by 15.375—not significantly different, but nonetheless different. The point that HyperCard is "easier" than BASIC, as speculated earlier, is supported to some extent by the HyperCard group's greater reduction in computer anxiety, especially if we subscribe to the notion that a reduction in computer anxiety reflects some degree of success at using the language; such success might be influenced by the ease of the language. It is supported to a much greater extent when comparing the two groups' performance measures, as explained next.

Performance Measures

The HyperCard group outperformed the BASIC group on both the programming and debugging exercises. When discussing, in the context of improving problem solving skills, the superior performance of the HyperCard group relative to the performance of the BASIC group, relatively greater ease when using HyperCard makes more sense. The HyperCard group debugged and programmed at a higher level than the BASIC group, but the BASIC group improved their problem solving skills and the HyperCard group did not. A logical interpretation is that HyperCard has simplified programming so much that HyperCard programmers can be more successful at using the language than those using BASIC—but, at the expense of "bypassing" critical thinking skills that promote the enhancement of problem solving skills.

Had the HyperCard group had similar performance scores as the BASIC group—which was not the case in this study—and also had not improved the problem solving skills—which was the case in this study—then our interpretation would have been that HyperCard is as difficult as BASIC—via the performance measures—and HyperCard programmers need more opportunities to use it in order to tap and develop problem solving skills. However, no improvement in problem solving skills along with significantly higher performance scores indicates that HyperCard is easier. Or, within the same amount of time, HyperCard programmers were more successful at using it than were those using BASIC—and apparently so much easier that HyperCard did not result in the programmers' improving their problem solving skills.

The relatively lower BASIC performance scores are consistent with two studies by Reed and Palumbo (1987/1988, 1992) which used identical BASIC performance measures. In the 1992 study the research-participants scored 81.07 on the programming task; this study, 82.08. For the debugging measure, the 1987/1988 participants scored 18.91; this study, 18.08. In all three studies, the samples were very similar and the instruments, where administered, were identical.

The Correlations

The two significant, negative correlations for the BASIC group provide even more evidence of BASIC's relatively greater difficulty; the correlations were (a) between debugging and

computer anxiety and (b) between programming and computer anxiety. Based on these correlations, it is clear that greater computer anxiety was associated with poorer debugging and programming performances. Although the negative relationship between computer anxiety and debugging emerged when including all the participants regardless of language grouping, the absence of a relationship for the HyperCard group, when looking at each group separately, limits the relationship just to the BASIC group. That is, computer anxiety had a significant, negative relationship with debugging in the multiple regression, when Course was held at its mean. When simple regressions were conducted to determine significant relationships within each group, the relationship between computer anxiety and debugging emerged when dealing with only the BASIC group but did not when dealing with only the HyperCard group. This occurrence within the BASIC group provides evidence that BASIC is more difficult because differing performance scores on language-related tasks were related to different levels of computer anxiety, as reflected by this negative relationship. The performance scores on the language-related tasks by the HyperCard group, however, were not related to different levels of computer anxiety, indicating that the performance tasks were not as difficult. A possible related example for illustration purposes is that differing levels of mathematics anxiety may not distinguish performance when the math problem is adding 2 plus 2. Most likely, people, regardless of their anxiety, could perform that calculation. However, if the math problem were more difficult—finding the square root of 27 times eight—varying levels of mathematics anxiety would distinguish performance. Our point is that the greater difficulty of BASIC, relative to HyperCard, is at least partly reflected by the negative, distinguishing relationships between computer anxiety and the two performance measures. There is, very likely, other evidence that indicates BASIC is more difficult, but it is not apparent in this study

IMPLICATIONS

The results of this study support our earlier suspicion that, while HyperCard is superior to BASIC for software construction, BASIC appears to be the language of choice when the purpose

is to develop problem solving skills. As we have stated, HyperCard is easier than BASIC, as reflected by the higher performance scores and greater reduction in computer anxiety. This point of greater ease for HyperCard is especially germane to pre-service and in-service instruction when prospective and practicing teachers are expected to work with a computer language—for example, as part of a computer literacy course. It may also be preferred over programming languages when students are to use a language to develop projects such as the ones described by Lehrer, Erikson, Love, and Connell (this issue) and by Reed and Rosenbluth (this issue). In these cases, the purpose of the instruction was to develop software or to construct knowledge structures, rather than develop problem solving skills. The absence of an increase in problem solving scores indicates that HyperCard should not be used if that is the purpose of instruction. BASIC apparently enhances problem solving skills partly because it is more difficult and requires more detailed thought.

A possible design change for further research might involve comparing BASIC commands with only HyperTalk scripting commands. In our study the HyperCard instruction involved seven sessions on authoring features and only three sessions on scripting commands. It must be noted, however, that many of the authoring features were facilitated by script-like aids, such as the navigation commands. Another consideration in this proposed study is whether or not HyperTalk might be taught without the foundational instruction on the authoring features. Given the probable need of this foundation, a research concern then would be time on task—that is, more time would need to be built into the instruction for the HyperCard group to provide them with this information. Would the result differences or similarities be due to the parallel features of the two languages or simply that the HyperCard group spent more time? Because time would be difficult to dispute, the instruction for the BASIC group would probably need to be re-thought in order to lengthen the exposure to BASIC so that the length of treatment for both groups would be similar. Perhaps the more "authoring" features of BASIC could be the central instruction early in the treatment.

If this lengthier HyperCard treatment were implemented, a repeated measure design might be used so that the mid-treatment data point would be the same as the post-treatment data in this

study. This would allow us to compare the pre-treatment/post-treatment scores of this study to the pre-treatment/mid-treatment scores of the future study. Similar findings would confirm or contradict the results in the HyperCard portion of this study.

Despite the changes we might make in our subsequent studies, it is clear via the ERIC search that there is a virtual vacuum of hypermedia/multimedia research. Probably almost any line of research would contribute. Fortunately, unlike some of the past research directions of instructional computing, there is a body of research—related both to CAI and programming languages—that can help formulate the research questions and guide the designs.

REFERENCES

Ashworth, D. & Stelovsky, J. (1989). Kanji City. An exploration of Hypermedia applications for CALL. *CALICO Journal, 6*(4), 27-39.

Barba, R. H. & Merchant, L. J. (1990). The effects of embedding generative cognitive strategies in science software. *Journal of Computers in Mathematics and Science Teaching, 10*(1), 59-65.

Bourne, D. E. (1990). Computer-assisted instruction, learning theory, and Hypermedia: An associative linkage. *Research Strategies, 8*(4), 160-171.

Bowers, D. & Chia, T. (1990). HyperCard in educational research: An introduction and case study. *Educational Technology, 30*(2), 19-24.

Boyle, C. C. & Snell, J. R. (1990). Intelligent navigation for semistructured hypertext documents. In R. McAleese & C. Green (Eds.), *Hypertext: State of the art* , (pp.29-42). Oxford, England: Intellect.

Brader, L. L. (1990). Tools of the courseware trade: A comparison of toolbook 1.0 and HyperCard 2.0. *Tech Trends, 35*(5), 10-17.

Camp, J. and Cogan, M. (1988). HyperCard: A milestone in educational computing. *Electronic Learning, March* (7), 46-47.

Campbell, R. (1989). (I learned it) through the Grapevine: Hypermedia at work in the classroom. *American Libraries, 20*(3), 200-202, 204-205.

Collourn, C. J. & Cockerton-Turner, T. (1990). Using hypertext for educational "help" facilities. In R. McAleese, & C. Green (Eds.), *Hypertext: State of the art*, (pp. 105-113). Intellect: Oxford, England.

Crane, G. & Mylonas, E. (1988). The perseus project: An interactive curriculum on classical Greek civilization. *Educational Technology, 28*(11), 25-32.

DiPardo, A. & DiPardo, M. (1990). Towards the metapersonal essay: Exploring the potential of hypertext in composition class. *Computers and Composition, 7*(3), 7-22.

Dufresne, A., Jolin, N. & Senteni, A. (1990). Hypertext documents for the learning of procedures. In R. McAleese & C. Green (Eds.), *Hypertext: State of the art*, (pp.97-104). Oxford, England: Intellect.

Duhrkopf, R. (1988). HyperCard as a tool for educational software development. *The American Biology Teacher, 50*(2), 121-122.

Friedlander, L. (1989). Moving Images into the classroom: Multimedia in higher education. *Laserdisk Professional, 2*(4), 33-38.

Gustafson K.L. & Reeves, T. C. (1990). IDioM: A platform for a course development expert system. *Educational Technology, 30*(3), 19-25.

Hall, W. Thorogood, P. Hutchings, G. & Carr, L. (1989). Using HyperCard and interactive video in education: An application in cell biology. *Educational & Training Technology International, 26*(3), 207-214.

Harris, M. & Cady, M. (1988). The dynamic process of creating hypertext literature. *Educational Technology, 28*(11), 33-40.

Holleman, P. (1990). Macintosh HyperCard library instruction program. *Community & Junior College Libraries, 7*(1), 59-75.

Hopper, J. (1990). Student-directed learning using HyperCard. *Computers in the Schools, 7*(4), 83-98.

Horton, S. V., Boone, R. A. & Lovitt, T. C. (1990). Teaching social studies to learning disabled high school students: effects of a hypertext study guide. *British Journal of Educational Technology, 21*(2), 118-131.

Jonassen, D. H. (1989). *Hypertext/Hypermedia.* Englewood Cliffs, NJ: Educational Technology Publications.

Jonassen, D. H. (1990). Semantic network elicitation: Tools for structuring hypertext. In R. McAleese & C. Green (Eds.), *Hypertext: State of the art*, (pp.142-154). Oxford, England: Intellect.

Lanza, A. & Roselli, T. (1991). Effects of the Hypertextual approach versus the structured approach on students' achievement. *Journal of Computer-Based Instruction, 18*(2), 48-50.

May, C. (1990). The American Short Story: From Poe to O. Henry. A HyperCard application. California State University, Long Beach, English Department. (ERIC Document Reproduction Service ED 330 315).

McDowell, E. E. (1989). The development of an instructional program on employment interviewing. Paper presented at the Annual Meeting of the Speech Communication Association (75th, San Francisco, CA, Novermber 18-21), (ERIC Document Reproduction Service ED 312 726).

McKnight, C. Dillon, A. & Richardson, J. (1990). A comparison of linear and hypertext formats in information retrieval. In R. McAleese & C. Green (Eds.), *Hypertext: State of the art*, (pp.11-19). Oxford, England: Intellect.

Miller, A. & Radziemski, C. (1988). Software reviews. *School Science and Mathematics, 88*(6), 528-530.

Monk, A. F. (1990). Getting to known locations in a hypertext. In R. McAleese & C. Green (Eds.), *Hypertext: State of the art*, (pp.21-27). Oxford, England: Intellect.

Neuman, D. (1991). Designing library instruction for undergraduates: Combining instructional systems design and naturalistic inquiry. *College and Research Libraries, 52*(2), 165-176.

Overbaugh, Richard C. & Reed, W. Michael. (1992). The effects of prior experience and instructional format on teacher education students' computer anxiety and performance. *Computers in the Schools.*

Palumbo, David B. & Reed, W. Michael. (1987/1988). Intensity of treatment and its relationship to programming languages/problem solving research. *Computers in the Schools, 4*(3/4), 113-121.

Palumbo, David. B. & Reed, W. Michael. (1991). The effects of BASIC programming on high school students' problem-solving skills. *Journal of Research on Computing in Education, 23*(3), 343-372.

Reed, W. Michael, Palumbo, David B., & Stolar, Aletha L. (1987/88). The comparative effects of BASIC and Logo instruction on problem solving skills. *Computers in the Schools, 4*(3/4), 99-111.

Reed, W. Michael & Palumbo, David B. (1987/88). The effect of BASIC programming on problem solving and computer anxiety. *Computers in the Schools, 4*(3/4), 85-97.

Reed, W. Michael & Palumbo, David B. (1992). The effect of BASIC instruction on problem solving skills over an extended period of time. *Journal of Educational Computing Research, 8*(3).

Percival, J.M. (1990). Graphic interfaces and oline information, *Online Review, 14*(1), 15-20.

Tripp, S.D. & Roby, W. (1990). Orientation and disorientation in a Hypertext lexicon. *Journal of Computer-Based Instruction, 17*(4), 120-124.

Tucker, S.A. & Dempsey, J. V. (1990). Photo-interviewing as a tool of inquiry. Paper presented at the National Consortium for Instruction and Cognition (Boston, MA, April 16-20), (ERIC Document Reproduction Service ED 318 800).

Underwood, J. (1989). HyperCard and interactive video. *CALICO Journal, 6*(3), 7-20.

Vaccaro, B. (1988). HyperCard applications. *Small computers in Libraries, June* (8), 30-35.

Appendix A

Programming Portion of BASIC Test

Write a program that---

asks the user his or her name
once the name is given, the screen should then clear
after clearing, the computer should address the user by name with a friendly message that is FLASHing
after a five-second delay, the program should ask the user for two numbers; the first question should appear halfway down the screen, and the other should appear a few lines above the bottom of the screen
after the two numbers are given by the user, the screen should clear and ask which mathematical operation he or she would like to perform on the two numbers—addition, subtraction, division, or multiplication
once the operation is selected, the screen should then clear and the "product" of the operation should appear (that is, if the user inputs 5 and 7 and chooses multiplication, then the "product" appearing on the screen would be 35)
after the "product" appears on the screen, the computer should then ask if the user would like to enter two new numbers; if so, you should have programmed in lines that would allow him to do that (the option to turn off the machine and run the program again is not acceptable!); if not, then the program should end

SAVE this program under your last name.

SOME POINTERS:

1. *Consider each part of the assignment separately; that is, break the task down into smaller subtasks.*
2. *For the division option, you need to include an error trap that does not allow 0 as the second number; simply put, you cannot divide by zero.*
3. *Error trap where you want the user to make a selection from a set of specific response choices.*
4. *Use REMs when you go from one section of the program to another.*

In this program you are using INPUT and GET commands, string and numeric variables, conditionals, iteration, some form of modularity, and various TEXT-related commands. You might want to check your program for this.

Appendix B

Programming Portion of HyperCard Test

Create a HyperCard stack using your name that---

consists of 4 cards with the first card as the title card
has at least two backgrounds with the title card having one background and the second card having another background;
contains pictures representing four different kinds of objects on the SECOND card (They can be from Art ideas or hand-drawn)
labels each object with a field or a tool.
has four transparent buttons on top of each of the objects. A click on a picture will display a description of the object. The description should be simple, one or two lines for example. The description of TWO objects will be displayed on CARDS and the description of the other two objects will be displayed on POP-UP FIELDS. There should be a return button on the description cards in order to go back. The return button can be on background if these two cards have a different background from the rest of cards (i.e. your stack has three backgrounds) or on foreground if they share the same background with the rest your stack has two backgrounds).
The pop-up fields will appear for 2 seconds, and then disappear.

Things to consider:

1. *The program should have an exit at any point.*
2. *The directions for the users as to what to do should be clear.*
3. *The movements between cards should be accompanied by visual effects.*
4. *Concentrate on the main part and keep a simple design for the backgrounds.*
5. *Make sure your stack works.*

Appendix C

Exercise for BASIC Debugging

Ten programming errors exist in the following ***high-res*** program. The errors range from incorrect sequence of commands to incorrect commands for a given mode to absent commands to unnecessary commands. These errors will result in an improperly executed program.

You need to do the following three steps:

1. identify the error
2. explain what would happen or not happen if the error were executed
3. give the correction.

```
10 HOME
20 GR
25 REM ******** BACKGROUND *******
30 HCOLOR = 1
40 FOR A = 0 TO 159
50 HPLOT 0, B TO 279, B
60 NEXT A
70 HCOLOR = 3
75 REM ******** ANIMATION *******
80 FOR PLANE = 0 TO 200 STEP -2
90 GOTO 1000
100 HCOLOR = 1
110 GOSUB 2000
120 HCOLOR = 0
130 GOSUB 1000
140 GOSUB 2000
150 NEXT PLAIN
1000 REM ********* FIRST PART OF OBJECT **************
1010 HPLOT PL, 0 TO PL + 20, 0 TO PL + 50, 10 TO PL + 20, 20 TO PL, 20 TO PL, 0
2000 REM ********* SECOND PART OF OBJECT **************
2010 HLIN PL, 21 TO PL + 80, 21 TO PL + 80, 31 TO PL, 21
2020 RETURN
```

Appendix D

Exercise for HyperCard Debugging

This stack has two cards. On the first card, there are a card field called "one" and a card button called "debug." On the second card there is a card field called "two," and a background field called "BK field." The debug button on card one has the script listed below. This script contains 10 errors.

You are to (a) identify the 10 errors; (b) explain why each is an error or what would happen if the program were executed without correcting it; and (c) correct them.

```
on mouseUp
    repeat 5
        hide field "one"
        visual barn door open very slow
        visual iris
        go next
        put "It is sunny." to card field "two"
        put 10 into holder
        put 20 into card field "BK field"
        add field "BK field"  holder
        put holder in msg
        go
        show card field "one"
        show card button "Show me"
    end
    do menu "Quit HyperCard"
end mouseUp
```